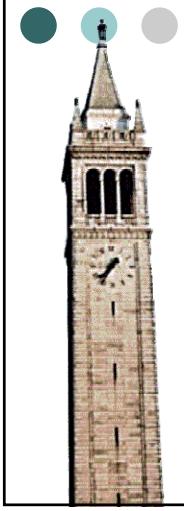


Overview of Research at CHESS

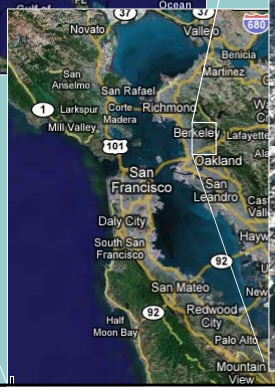
Center for Hybrid and Embedded Software Systems

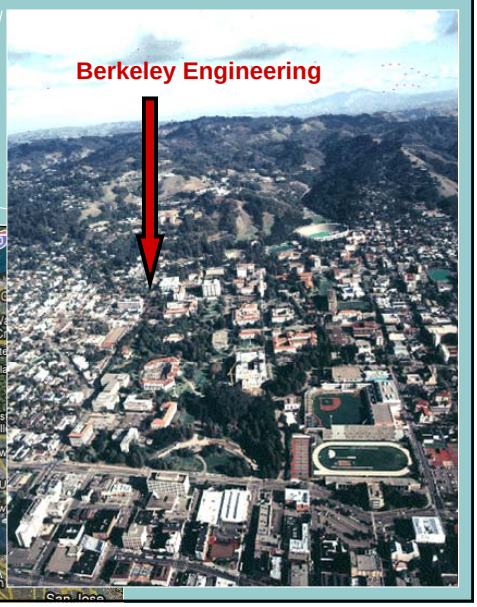
Stavros Tripakis

Associate Researcher
UC Berkeley
stavros@eecs.berkeley.edu

Presented to IHI Delegates
Feb 20, 2013
Berkeley, CA

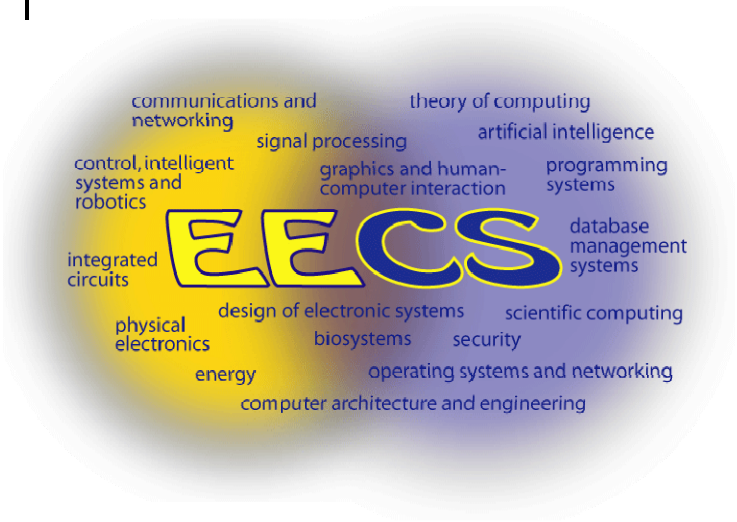
University of California at Berkeley





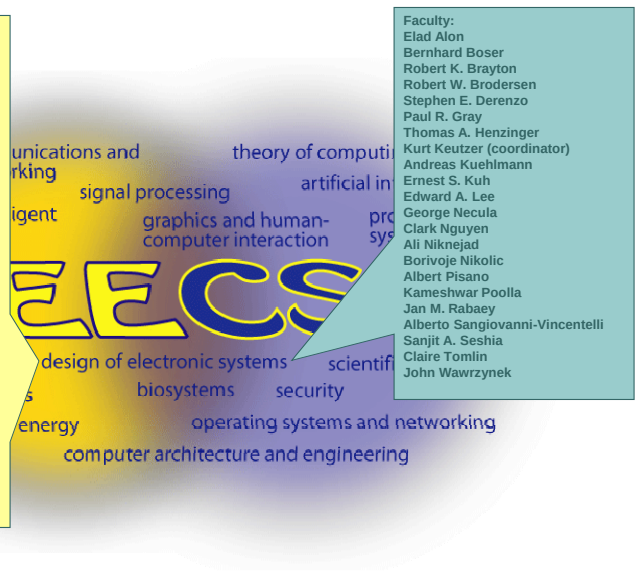
UC Berkeley has one of the best public engineering schools in the world.

Overview of EECS at Berkeley



Design Sciences Research Area

- ***Electronic Design Automation***
  Algorithms and techniques to support computer-aided design and optimization of complex hardware and software systems.
- ***Embedded Software Systems***
  Models of computation, specification languages, real-time systems, hardware and software synthesis and compilation for electronic systems.
- ***Modeling and Verification***
  Models of hardware and software systems together with analysis techniques that identify design flaws, performance problems, and vulnerabilities.



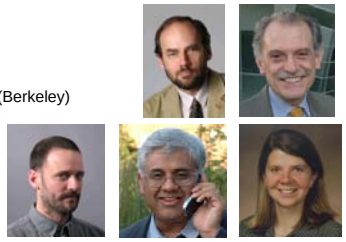
Faculty:
Elad Alon
Bernhard Boser
Robert K. Brayton
Robert W. Brodersen
Stephen E. Derenzo
Paul R. Gray
Thomas A. Henzinger
Kurt Keutzer (coordinator)
Andreas Kuehlmann
Ernest S. Kuh
Edward A. Lee
George Necula
Clark Nguyen
Ali Niknejad
Borivoje Nikolic
Albert Pisano
Kameshwar Poolla
Jan M. Rabaey
Alberto Sangiovanni-Vincentelli
Sanjit A. Seshia
Claire Tomlin
John Wawrzynek

CHESS: Center for Hybrid and Embedded Software Systems

Principal Investigators
- Thomas Henzinger (EPFL)
- Edward A. Lee (Berkeley)
- Alberto Sangiovanni-Vincentelli (Berkeley)
- Shankar Sastry (Berkeley)
- Janos Sztipanovits (Vanderbilt)
- Claire Tomlin (Berkeley)

Executive Director
- Christopher Brooks

Associated Faculty
- David Auslander (Berkeley, ME)
- Ahmad Bahai (Berkeley)
- Ruzena Bajcsy (Berkeley)
- Gautam Biswas (Vanderbilt)
- Ras Bodik (Berkeley, CS)
- Bella Bollobas (Memphis)
- Karl Hedrick (Berkeley, ME)
- Gabor Karsai (Vanderbilt)
- Kurt Keutzer (Berkeley)
- George Necula (Berkeley, CS)
- Koushik Sen (Berkeley, CS)
- Sanjit Seshia (Berkeley)
- Jonathan Sprinkle (Arizona)
- Masayoshi Tomizuka (Berkeley, ME)
- Pravin Varaiya (Berkeley)



the Berkeley directors of Chess

This center, founded in 2002, blends systems theorists and application domain experts with software technologists and computer scientists.

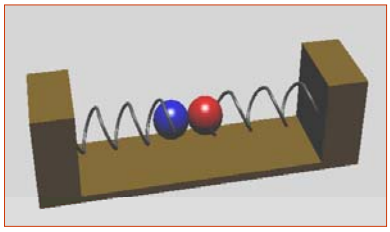
Some Research Projects
- Precision-timed (PRET) machines
- Distributed real-time computing
- Systems of systems
- Theoretical foundations of CPS
- Hybrid systems
- Design technologies
- Verification
- Intelligent control
- Modeling and simulation

Applications
- Air traffic control
- Avionics
- Automotive
- Building systems
- Factory automation
- Instrumentation
- Medical systems
- Process control
- Synthetic biology
- Test & measurement

Tripakis, Berkeley 5

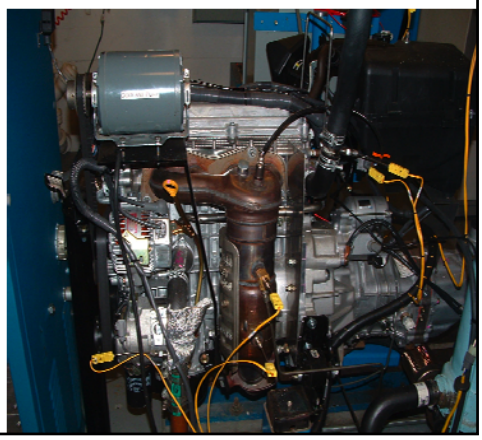
Hybrid Systems
Where it started

A model of a spring-mass system with collisions, modeled in Ptolemy II:



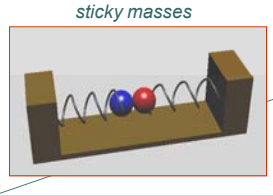
Consider the velocity of each mass. Is it continuous? What about the acceleration?

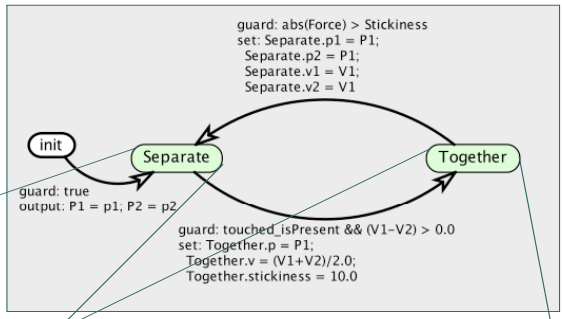
Toyota test cell at Berkeley used to experiment with hybrid systems models.

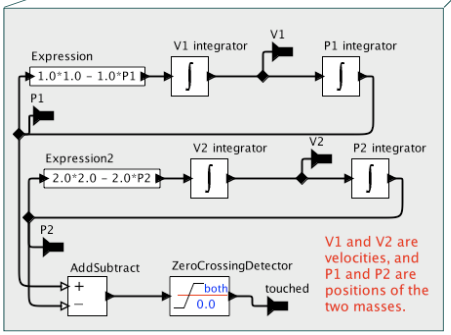


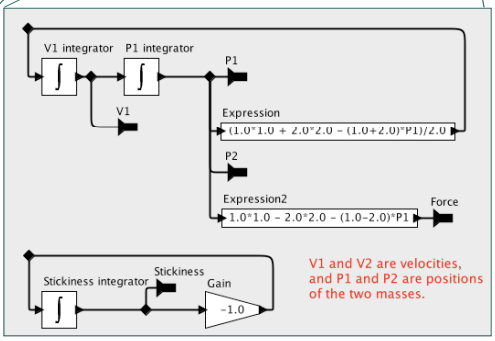
Hybrid automata = state machines + differential equations

sticky masses









Tripakis, Berkeley 7

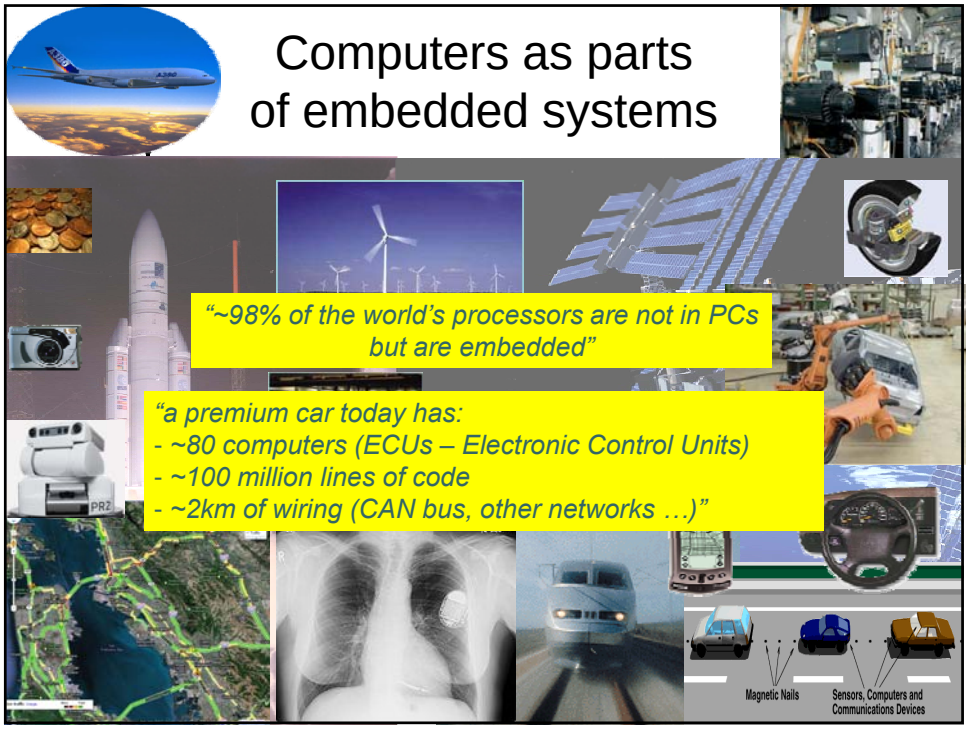
Cyber-Physical Systems (CPS)

Where it is going

CPS: Orchestrating networked computational resources with physical systems.

Tripakis, Berkeley 8

Computers as parts of embedded systems

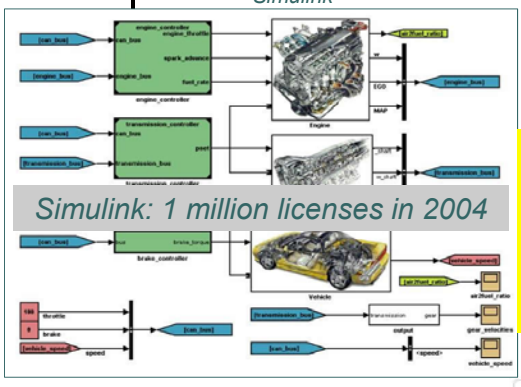


"~98% of the world's processors are not in PCs but are embedded"

"a premium car today has:
- ~80 computers (ECUs – Electronic Control Units)
- ~100 million lines of code
- ~2km of wiring (CAN bus, other networks …)"

Embedded system languages & tools

Simulink



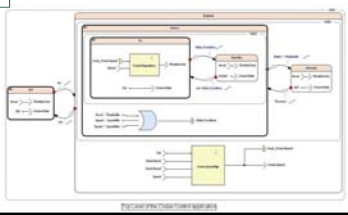
Simulink: 1 million licenses in 2004

Key concepts:
- reactive behavior
- concurrency
- timing
- I/O
- …

Modelica / Dymola

LabVIEW

Key capabilities:
- simulation
- code generation
- verification

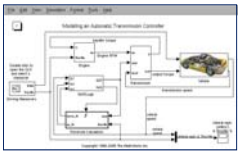


SCADE

keley 10

Vision

These modeling languages of today will become the **system-programming** languages of tomorrow



Rich input languages: concurrency, time, robustness, reliability, energy, security, …

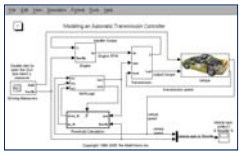
system compiler

Complex execution platforms: networked, distributed, multicore, …

Powerful analyses: model-checking, WCET analysis, schedulability, performance analysis, reliability analysis, …

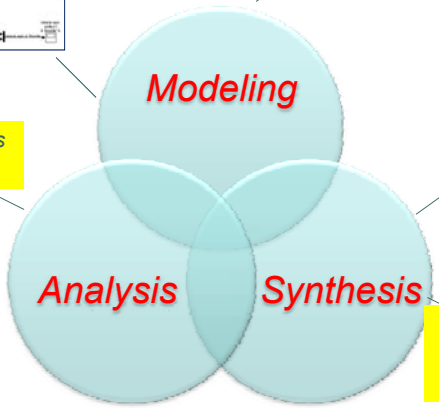
Tripakis, Berkeley 11

Model-Based Design



How to describe what we want?

How to be sure that this is what we want?

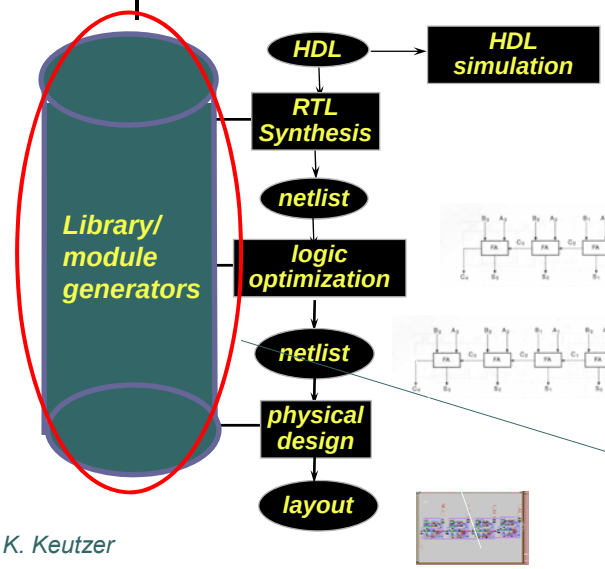


How to build it? Automatically Correct-by-construction

Tripakis, Berkeley 12

Is Model-Based Design a realistic vision?

Instance of a successful and model-based design flow



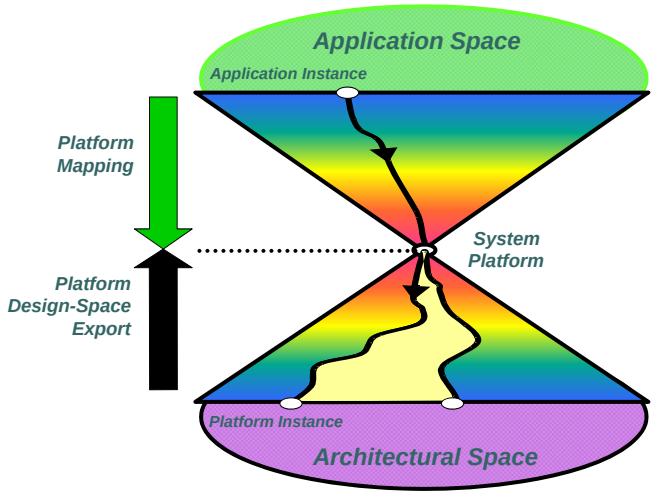

```
module Full_Adder_4 (sum, c_out, a, b, c_in);
 output      [3:0]sum;
 output      c_out;
 input       [3:0] a, b;
 input       c_in;
 wire        c_in2, c_in3, c_in4;
 Full_adder  M1 (sum[0], c_in2, a[0], b[0], c_in);
 Full_adder  M2 (sum[1], c_in3, a[1], b[1], c_in2);
 Full_adder  M3 (sum[2], c_in4, a[2], b[2], c_in3);
 Full_adder  M4 (sum[3], c_out, a[3], b[3], c_in4);

endmodule
```


A lot of this technology was invented here!

The importance of libraries not to be underestimated!

K. Keutzer

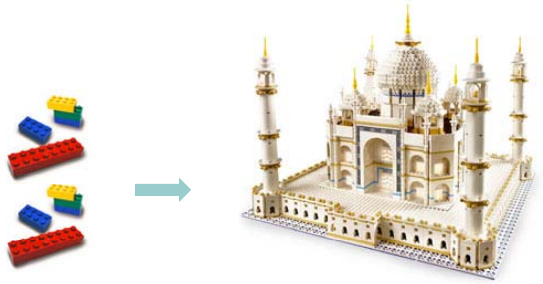
Platform-based design



Alberto Sangiovanni-Vincentelli

Tripakis, Berkeley 15

Component-based design



Components *System*

Tripakis, Berkeley 16

Agenda

- Some research activities going on at CHESS: this talk, until 10:30am (Stavros)
- Tour of CHESS facilities, discuss with students and researchers, look at selected projects: 10:30am – noon
  - 10:35am - 10:50am: David Broman, Michael Zimmer – PRET
  - 10:50am - 11:05am: Patricia Derler - Ptides
  - 11:05am - 11:20am: Alex Donze - Hybrid system verification
  - 11:20am - 11:35am: Ben Zheng - Terraswarm and localization
  - 11:35am - 11:50am: Mehdi Maasoumy - Smart/green buildings
  - 11:50am - noon: Sayak Ray - HW verification and synthesis

Some of the CPS Research in Chess

- *Foundations:* Heterogeneous modeling with actors.
- *Bottom up*: Embedded processors (PRET).
- *Top down*: Distributed real-time systems (PTIDES).
- *Holistic*: Scalable model-based design.

The Ptolemy project

The Ptolemy project studies modeling, simulation, and design of concurrent, real-time, embedded systems. The focus is on assembly of concurrent components. The key underlying principle in the project is the use of well-defined models of computation that govern the interaction between components. A major problem area being addressed is the use of heterogeneous mixtures of models of computation. A software system called Ptolemy II is being constructed in Java, and serves as the principal laboratory for experimentation.

Tripakis, Berkeley 19

Contributors to Ptolemy II

Principal Authors

- Christopher Brooks
- Dai Bui
- Chamberlain Fong
- John Davis, II
- Patricia Derler
- Thomas Huining Feng
- Mudit Goel
- Rowland Johnson
- Bilung Lee
- Edward Lee
- Ben Lickly
- Jie Liu
- Xiaojun Liu
- Lukito Muliadi
- Stephen Neuendorffer
- John Reekie
- Neil Smyth
- Jeff Tsay
- Yuhong Xiong
- Haiyang Zheng
- Gang Zhou

Other Contributors

- *Jim Armstrong*
- *Vincent Arnould*
- *Kyungmin Bae*
- *Philip Baldwin*
- *Chad Berkley*
- *Frederic Boulanger*
- *Raymond Cardillo*
- *Jannette Cardoso*
- *Adam Cataldo*
- *Christine Cavanessians*
- *Chris Chang*
- *Albert Chen*
- *Chihong Patrick Cheng*
- *Elaine Cheong*
- *Colin Cochran*
- *Brieuc Desoutter*
- *Pedro Domecq*
- *William Douglas*
- *Johan Eker*
- *Thomas Huining Feng*
- *Tobin Fricke*
- *Teale Fristoe*
- *Shanna-Shaye Forbes*
- *Hauke Fuhrmann*
- *Geroncio Galicia*
- *Ben Horowitz*
- *Heloise Hse*
- *Efrat Jaeger*
- *Jörn Janneck*
- *Zoltan Kemenczy*
- *Bart Kienhuis*
- *Christoph Meyer Kirsch*
- *Sanjeev Kohli*
- *Vinay Krishnan*
- *Robert Kroeger*
- *Daniel Lázaro Cuadrado*
- *David Lee*
- *Man-kit (Jackie) Leung*
- *Michael Leung*
- *John Li*
- *Isaac Liu*
- *Andrew Mihal*
- *Eleftherios Matsikoudis*
- *Aleksandar Necakov*
- *Mike Kofi Okyere*
- *Sarah Packman*
- *Shankar Rao*
- *Bert Rodiers*
- *Rakesh Reddy*
- *Adriana Ricchiuti*
- *Sonia Sachs*
- *Ismael M. Sarmiento*
- *Michael Shilman*
- *Sean Simmons*
- *Mandeep Singh*
- *Miro Spoenemann*
- *Peter N. Steinmetz*
- *Dick Stevens*
- *Mary Stewart*
- *Ned Stoffel*
- *Manda Sutijono*
- *Stavros Tripakis*
- *Neil Turner*
- *Guillaume Vibert*
- *Kees Vissers*
- *Brian K. Vogel*
- *Yuke Wang*
- *Xavier Warzee*
- *Scott Weber*
- *Paul Whitaker*
- *Winthrop Williams*
- *Ed Willink*
- *Michael Wirthlin*
- *Michael Wetter*
- *William Wu*
- *Xiaowen Xin*
- *Paul Yang*
- *James Yeh*
- *Nick Zamora*
- *Charlie Zhong*

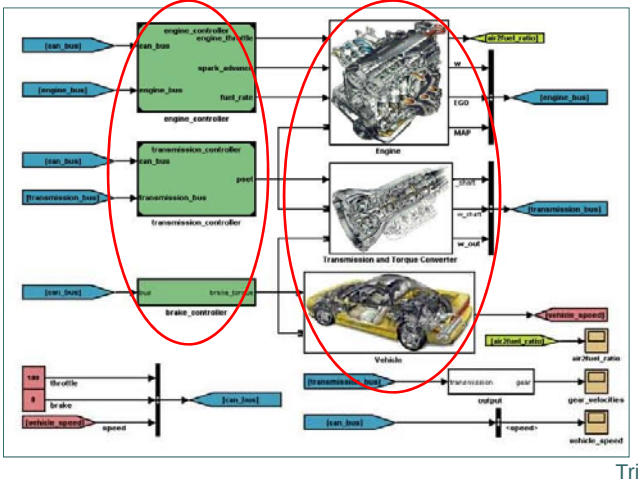
Lee, Berkeley 4

Heterogeneous modeling

- One system, many models!
- Different parts of the same system typically modeled using different types of models.
- How to combine these different models?
- How to reason about the system as a whole?

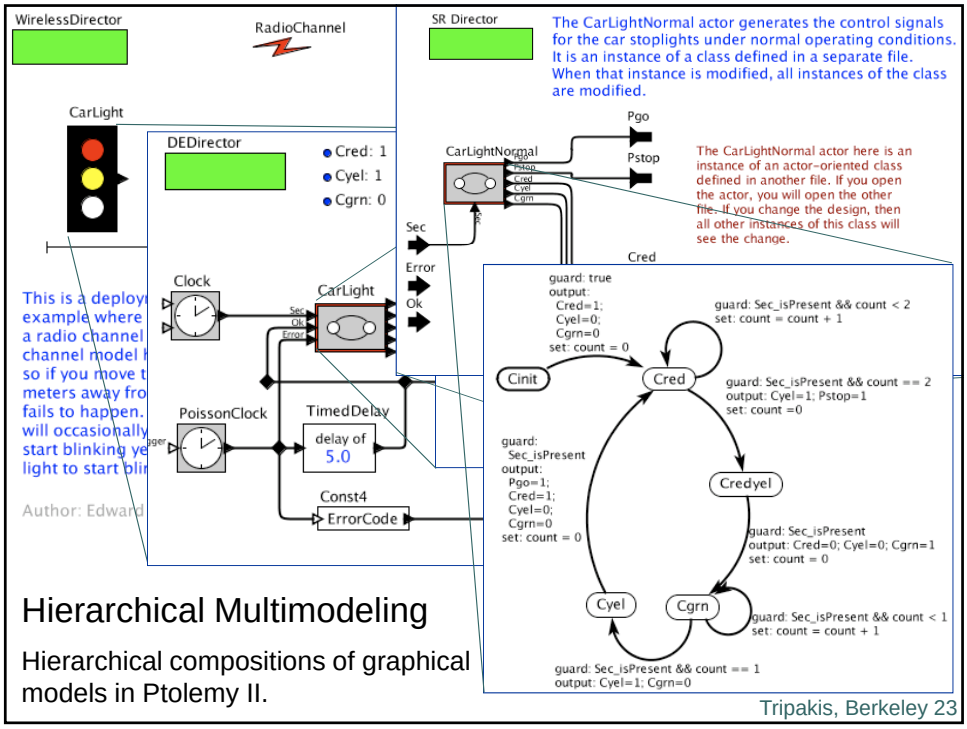
Examples of heterogeneous models

- Control systems = discrete + continuous dynamics



Simulink

Copyright The Mathworks



Hierarchical Multimodeling

Hierarchical compositions of graphical models in Ptolemy II.

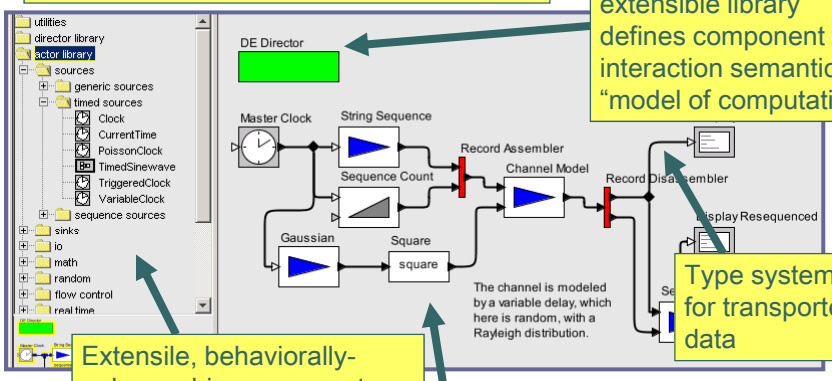
Ptolemy = Actors + Directors + …

- Actors: basic functionality (“atomic” blocks)
- Directors: simulation engines
  - Give meaning to the composition of actors
  - Different engines implement different models of concurrency (synchronous, asynchronous, discrete, continuous, …)
  - (almost) Arbitrary hierarchical compositions
- + GUI + type system + …

Ptolemy II: a Laboratory for Actor-Oriented Models of Computation

Concurrency management supporting dynamic model structure.

Director from an extensible library defines component interaction semantics or "model of computation."

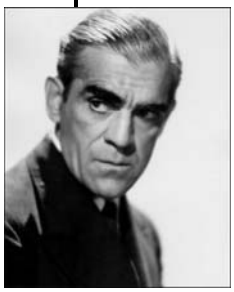


Type system for transported data

Extensile, behaviorally-polymorphic component library.

Visual editor supporting an abstract syntax

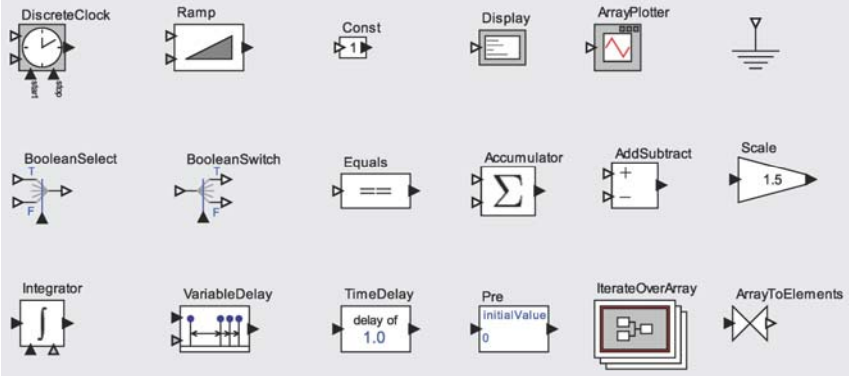
Actors



Actors

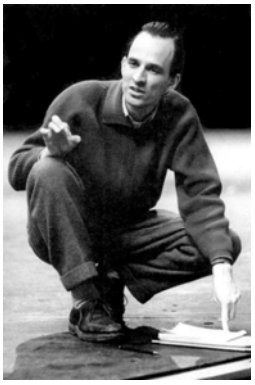
- Ptolemy has a library of predefined actors
- They are all Java classes that implement a standard interface called "*executable*"

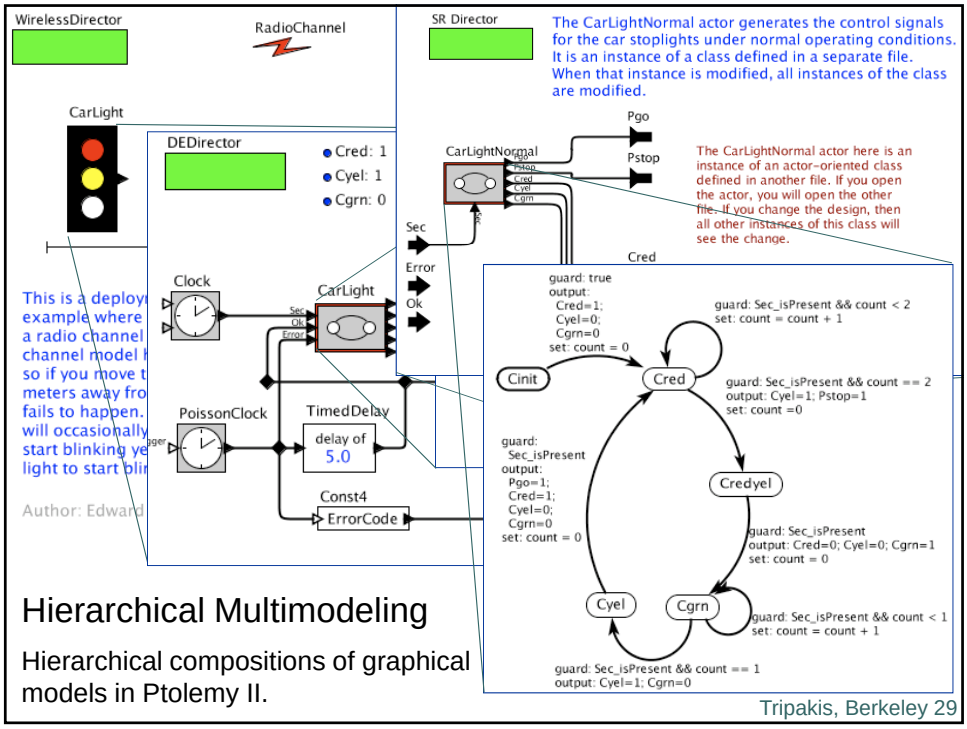
Some actors



Tripakis, Berkeley 27

Directors





Hierarchical Multimodeling

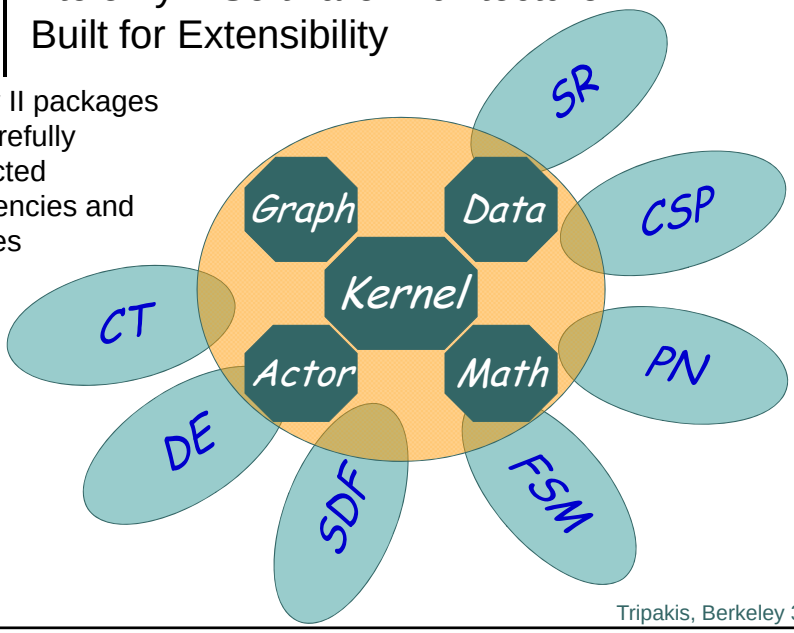
Hierarchical compositions of graphical models in Ptolemy II.

Models of Computation Implemented in Ptolemy II

- CI – Push/pull component interaction
- Click – Push/pull with method invocation
- CSP – concurrent threads with rendezvous
- Continuous – continuous-time modeling with fixed-point semantics
- CT – continuous-time modeling
- DDF – Dynamic dataflow
- DE – discrete-event systems
- DDE – distributed discrete events
- DPN – distributed process networks
- FSM – finite state machines
- DT – discrete time (cycle driven)
- Giotto – synchronous periodic
- GR – 3-D graphics
- PN – process networks
- Rendezvous – extension of CSP
- SDF – synchronous dataflow
- SR – synchronous/reactive
- TM – timed multitasking
- …

Ptolemy II Software Architecture Built for Extensibility

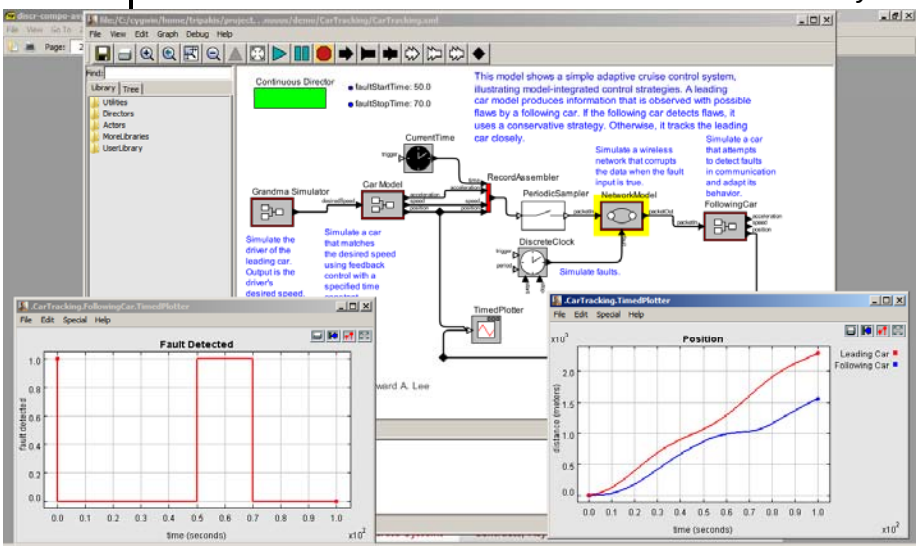
Ptolemy II packages have carefully constructed dependencies and interfaces



Tripakis, Berkeley 31

Brief Ptolemy demo

A car-tracking controller, modeled in Ptolemy II



Tripakis, Berkeley 32

Some of the CPS Research in Chess

- *Foundations:* Timed computational semantics.
- *Bottom up*: Embedded processors (PRET).
- *Top down*: Distributed real-time systems (PTIDES).
- *Holistic*: Scalable model-based design.

Bottom Up: Embedded Processors

Precision-Timed (PRET) Machines

Make temporal behavior as important as logical function.

Timing precision with performance: Challenges:

- Memory hierarchy (scratchpads?)
- Deep pipelines (interleaving?)
- ISAs with timing (deadline instructions?)
- Multicore (dedicated I/O & real-time processors?)
- Predictable memory management (Metronome?)
- Languages with timing (discrete events? Giotto?)
- Predictable concurrency (synchronous languages?)
- Composable timed components (actor-oriented?)
- Precision networks (TTA? Time synchronization?)

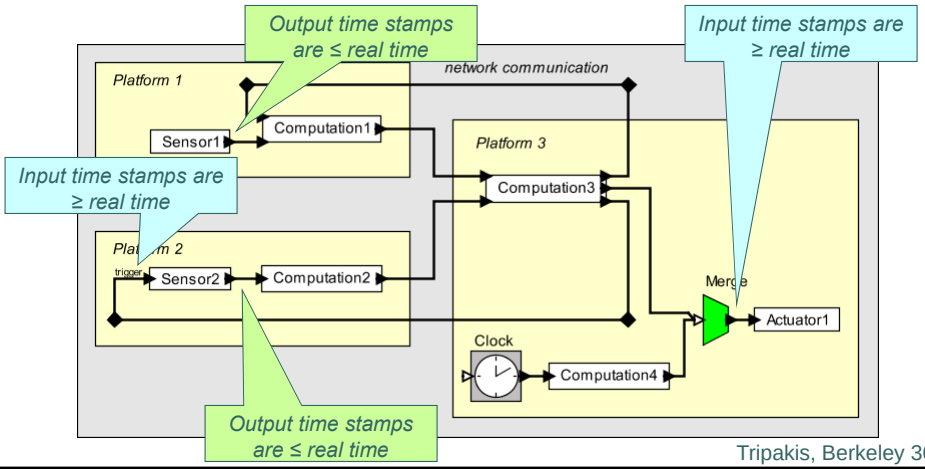
See S. Edwards and E. A. Lee, "**The Case for the Precision Timed (PRET) Machine**," in the *Wild and Crazy Ideas* Track of the *Design Automation Conference* (DAC), June 2007.

Some of the CPS Research in Chess

- *Foundations:* Timed computational semantics.
- *Bottom up*: Embedded processors (PRET).
- *Top down*: Distributed real-time systems (PTIDES).
- *Holistic*: Scalable model-based design.

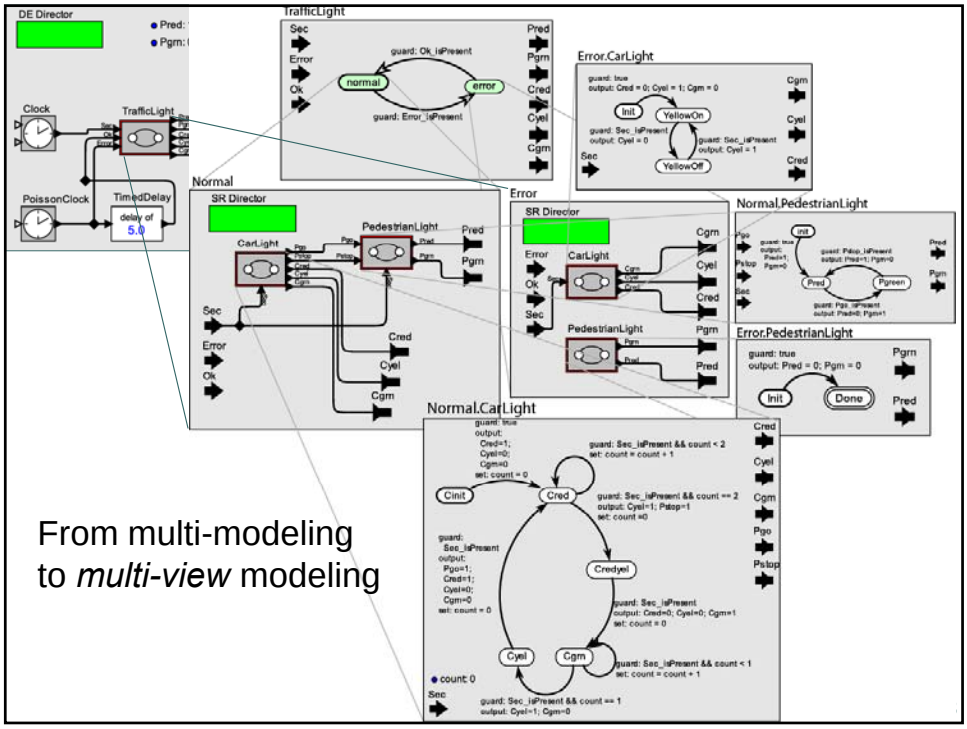
PTIDES: Programming Temporally Integrated Distributed Embedded Systems

Distributed execution under DE semantics, with "model time" and "real time" bound at sensors and actuators.



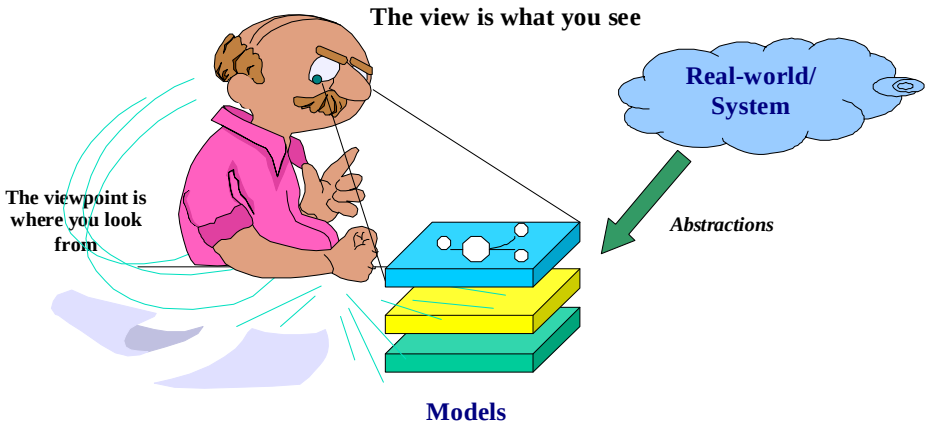
Some of the CPS Research in Chess

- *Foundations:* Timed computational semantics.
- *Bottom up*: Embedded processors (PRET).
- *Top down*: Distributed real-time systems (PTIDES).
- *Holistic*: Scalable model-based design.



From multi-modeling
to *multi-view* modeling

Viewpoints, views and models

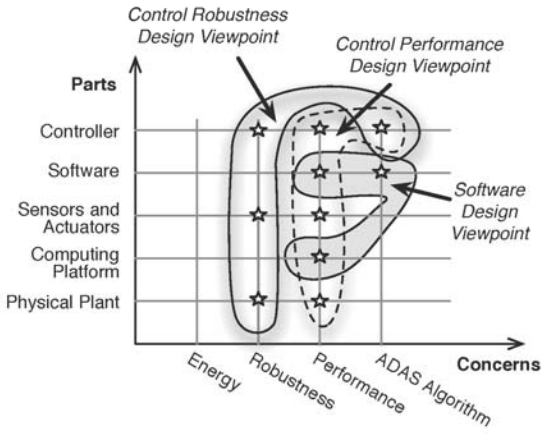


Martin Törngren

Multiple viewpoints

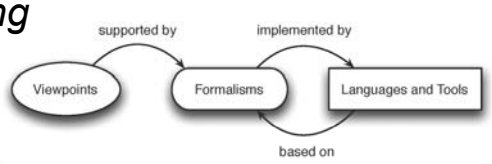
Stakeholder/Role	Concerns	Analysis/Synthesis	Model Characteristics
Electrical engineer/hardware architecture	ECU interfaces and EMC	Electrical load and tests	Logic, continuous, and FEM
Software engineer (body area)	Logics of functionality	Simulation of behavior	Discrete-event
Quality engineer	Reliability	Life-time prediction and FMEA	Stochastic and logic (e.g., failure analysis)
Mechanical engineer	ECU packaging, geometry, and fitting	Cable length and geometry alignment	2D and 3D mechanical
Cost controller	Product cost	Profitability and sensitivity analysis	Economical and uncertainty explicit
Integration engineer	Verification communications and distributed functions	Testing! Automation of tests and generation of test documentation	Discrete-event (test cases) and logical structures (e.g., configurations)
Safety engineer	System safety	FTA and FMEA	Logic, discrete-event, and stochastic
Control system engineer	Performance and robustness, and disturbances	Behavior simulation, robustness analysis, and controller synthesis	Continuous-time, discrete-time, and discrete-event
Thermoanalyst	Temperature	Heat transfer	FEM, etc.

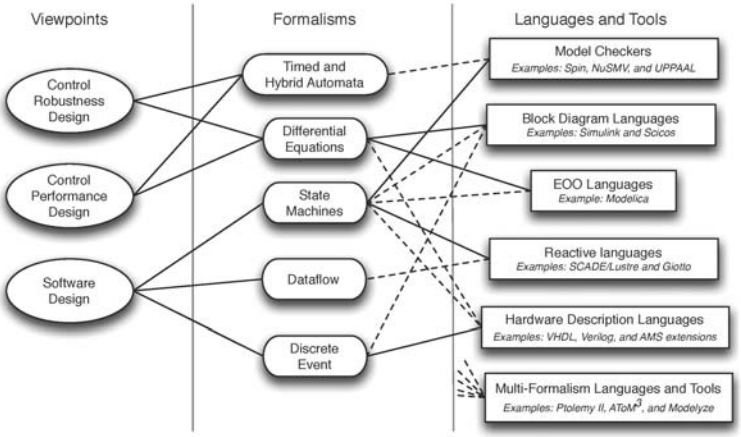
Multiple viewpoints



Tripakis, Berkeley 41

A framework to support stakeholders - *focus on modeling*





Berkeley 42

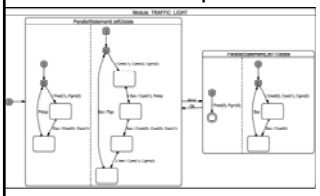
Multiple views (projections), with partially overlapping and related content

Martin Törngren

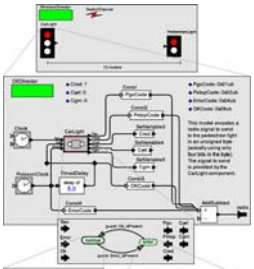
Tripakis, Berkeley 43

Multi-View Modeling:

Distinct and separate models of the same system are constructed to model different aspects of the system.

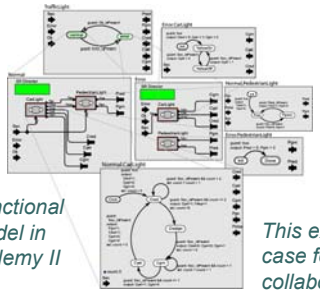


Functional model in Statecharts



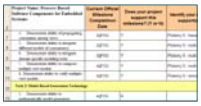
Deployment model in Ptolemy II

Verification model in SMV



Functional model in Ptolemy II

This example is a test case for a collaborative project with Lockheed-Martin



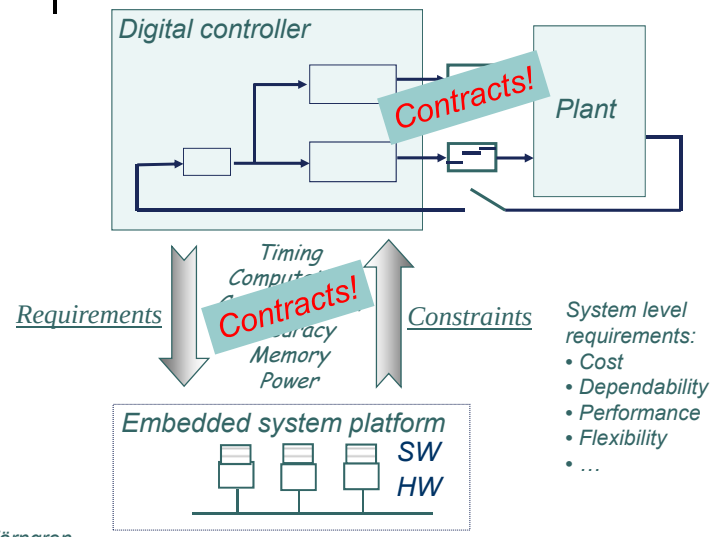
Reliability model in Excel

Tripakis, Berkeley 44

Multi-view modeling

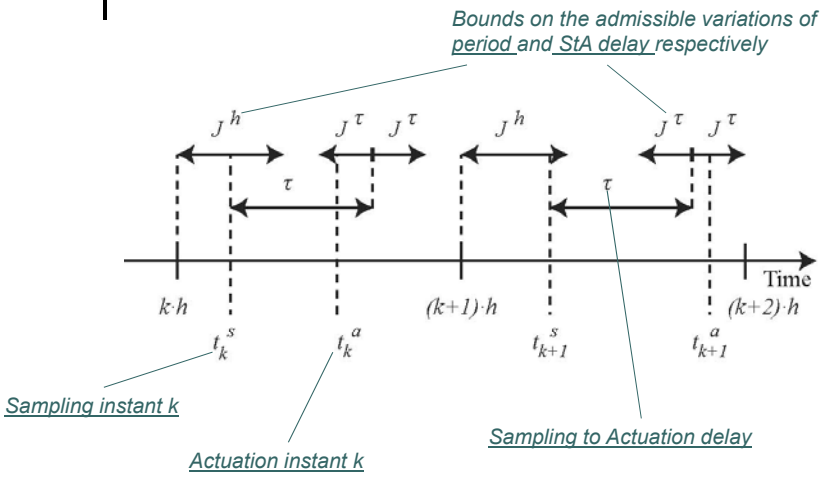
- Example concrete problem:
  - How to maintain consistency between AADL Behavior annex and Error annex models?

Design Contracts



Martin Törngren

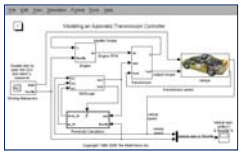
Example timing contract for control



Martin Törngren

Tripakis, Berkeley 47

Compositional Model-Based Design



How to create large models from libraries of submodels?

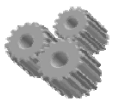
Modeling

How to analyze submodels separately and derive global system properties?

Compositionality, Modularity

Analysis

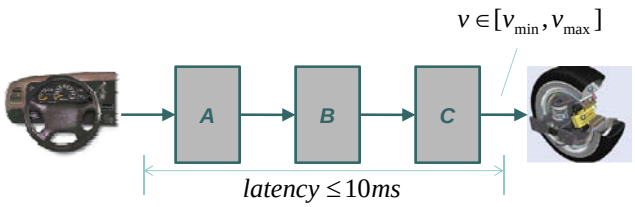
Synthesis



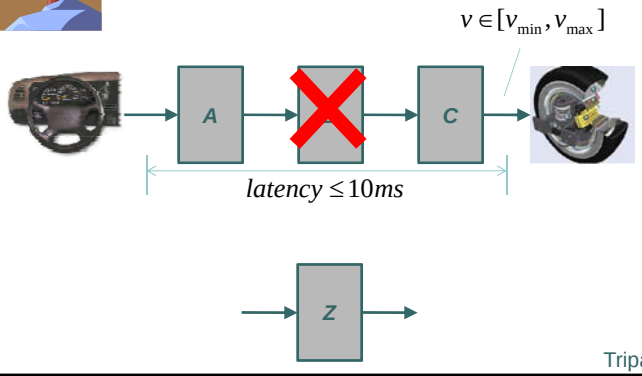
How to implement different parts of the system independently?

Tripakis, Berkeley 48

Substitutability

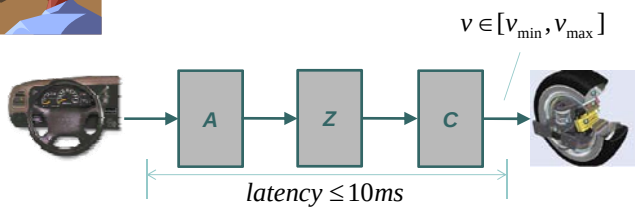


Substitutability

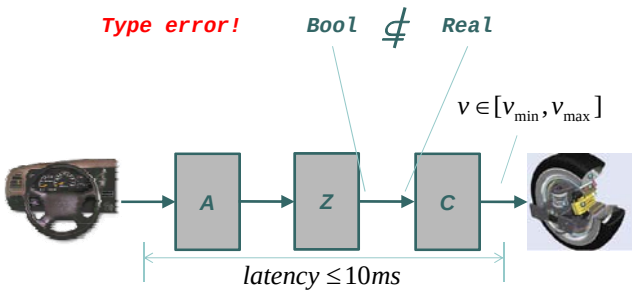


Substitutability

How to ensure properties are preserved?



Interface theories = **behavioral** type theories



Which type theories for safety & performance properties?

Research as a Business

Our Business Model:

Maximize the impact of our work, and the rest follows

Chess Industrial Partnerships

- Close interaction between academic research and industrial experience, often involving frequent deep technical interactions.
- Facilitate the creation and transfer of modern, "new economy" software technology methods and tools to "old economy" market sectors.
- Focus on industries where embedded software plays a central role, such as:
  - aerospace
  - automotive
  - Instrumentation, test, and measurement

Options for Funding Research

- Gifts
- Quasi-gifts (CADCAM, BEECSA, ...)
- Center memberships
- Contracts

Center memberships have proved far more effective than the alternatives.

Intellectual Property in CHESS

Commitment is:

- Software will be open source
- Patents will be rare

IP Principles

- Address researcher concerns
  - Maximize the impact of the work
  - Freedom to publish papers
  - Freedom to release open source software
  - No obligation to patent results
  - Ability to patent results
  - Minimal obstacles to commercialization
- Address company concerns:
  - Value for the money
  - Protection from being “locked out” of research results
  - Viable VIF agreements
  - Desire to “expense” contributions

Intellectual Property Rights

From Appendix C of the CHESS Agreement

- “The objective of CHESS is to maximize the impact of its research. To achieve this, CHESS will maintain an open atmosphere that encourages early and frequent publication and other public dissemination of research results. Software will primarily be released using an open source license such as the Berkeley Software Distribution (BSD) License. Selected software, such as those subject to third party obligations, may be released under different licenses.”
- “Patents are expected to be rare. . .”

The BSD License: Used for most software

Copyright (c) *YEAR* The Regents of the University of California.
All rights reserved.

Permission is hereby granted, without written agreement and without license or royalty fees, to use, copy, modify, and distribute this software and its documentation for any purpose, provided that the above copyright notice and the following two paragraphs appear in all copies of this software.

IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

Spinoffs of Ptolemy Software

- **Agilent ADS**: Leading design software for analog, RF, and mixed-signal design, primarily targeted to wireless systems development, based on Ptolemy Classic.
- **Cal actor design language**, adopted by MPEG for specification of coding standards, and used by Thales, Xilinx, and others for FPGA and multicore software design.
- **iSencia Passerelle** is based on Ptolemy II and is used to prepare experiments for the Soleil synchrotron.
- **Kepler**: A System for Scientific Workflows, is a cross-project collaboration to develop open source tools for Scientific Workflows and is based on the Ptolemy II.
- **Mirabilis Design VisualSim**, built on top of Ptolemy II, does performance analysis and system architecture, rapid system modeling and hardware/software tradeoff analysis.
- **ML Design Technologies' MLDesigner**, is a platform that leverages Ptolemy Classic for modeling and analyzing the architecture, function and performance of high level system.
- **And more**: Boeing, Bosch, Cadence, Lockheed Martin, Research in Motion, Thales (ArrayOL), VPI Systems, White Eagle Technologies, etc.

CHESS Industrial Membership Levels

Affiliate > $75k/year	Affiliate membership includes the following benefits: • Invitation to periodic reviews of CHESS. • Access to selected internal CHESS websites. • Access to publications, reports and presentations by CHESS researchers. • Access to students and faculty in CHESS. • An annual research report of the activities of CHESS. • Advance notice of intellectual property created by CHESS. • Intellectual property access as defined in the agreement. (See Appendix C(1))
Small or Minority-Owned Business > $10k/year	All of the benefits of an Affiliate
Partner > $150k/year	All of the benefits of an Affiliate, plus the following: • Opportunity to place visitors, as Visiting Industrial Fellows (VIF) at the University. • Selected early access to software developed by CHESS.
Premium Partner > $300k/year	All of the benefits of a Partner, plus the following: • Intellectual property access as defined in the agreement. • Upon request, an annual private research review meeting at the University or at a mutually agreeable site.

Conclusion

Chess has a rich portfolio of research projects and an established track record of working effectively with industry to maximize the impact of the work.

Agenda

- Some research activities going on at CHESS: this talk, until 10:30am (Stavros)
- Tour of CHESS facilities, discuss with students and researchers, look at selected projects: 10:30am – noon
  - 10:35am - 10:50am: David Broman, Michael Zimmer – PRET
  - 10:50am - 11:05am: Patricia Derler - Ptides
  - 11:05am - 11:20am: Alex Donze - Hybrid system verification
  - 11:20am - 11:35am: Ben Zheng - Terraswarm and localization
  - 11:35am - 11:50am: Mehdi Maasoumy - Smart/green buildings
  - 11:50am - noon: Sayak Ray - HW verification and synthesis